AMENDMENTS TO THE SPECIFICATION:

In the latest Office Action and during the telephone interview dated June 25, 2009, the Examiner requested that additional specification details from co-pending applications be incorporated into the specification to support the claim amendments also requested by the Examiner.

Accordingly, the five paragraphs, previously incorporated by reference from these co-pending applications and variously amended, are now further revised, as follows, to incorporate additional descriptions. These following sentences are intended to be inserted immediately preceding the subtitle "Level 3 Prefetching of Kernel Routines" on page 12 of the specification:

The present invention includes using data stored in non-standard format, including, more particularly, the non-standard format described in co-pending application 10/671,888, referred to herein as "register block" format.

The present invention also is directed to Single Instruction, Multiple Data (SIMD) machines, where $k > 1$ indicates a number of data capable of being simultaneously moved in a single instruction. Thus, in the example described in the third of the above-identified co-pending applications, wherein the register block format was demonstrated using a 2-by-2 block, referred to therein as a "pseudo-matrix", ~~k = 4~~ k = 2.

The register block data format exemplarily used in the present invention involves blocks of matrix data of size p-by-q where p and q are small integers so that the pieces of these blocks can be fitted into the registers of a particular architecture to achieve a desirable data format stored in these registers. The layout of these blocks is arbitrary. In usual cases, the p-by-q sub-blocks will be laid out either in row- or column-major format. But a key idea is that the arbitrary layout of these blocks is tailored to the architectural design of the FPU and its associated floating point registers.

All modern programming languages (C, Fortran, etc.) store matrices in two-

dimensional arrays. That is, let matrix A have M rows and N columns. The standard column major format of A is as follows.

Each of the N columns of A is stored as a contiguous vector (stride 1). Each of the M rows of A is stored with consecutive elements separated by LDA (Leading Dimension of A) storage locations (Stride LDA). Let A(0,0) be stored in memory location α. The matrix element A(i,j) is stored in memory location $\alpha + i + LDA*j$. It is important to note here that stride 1 is optimal for memory accesses and that stride LDA is poor. Also, almost all level 3 linear algebra code treats rows and columns about equally.

As further explained in the sixth and seventh of the co-pending applications, there is also an aspect in the present invention of providing a technique in which one of six possible kernel types is selectively available, based on which of the six kernel types allows a matrix (or sub matrix, depending upon the size of the matrix) to best fit into a cache (e.g., an L1 cache). The importance of having six kernel types available is that stride one memory access is desirable for matrix processing. The matrices or submatrices A, B, and C are usually stored either by row or by column or in register block format.

By having six kernel types, one can choose a kernel in which stride one is available for both, or possibly three, operands. If only one, instead of six, kernel types are available, data copying must be done to provide the format of the single kernel conventionally used. This means a certain performance loss that might even have to be repeated several times during the processing.

The importance of having six kernel types available is that stride one memory access is desirable for matrix processing. The matrices or submatrices A and B are usually stored either by row or by column. By having six kernel types, one can choose a kernel in which stride one is available for both operands. Having only one kernel type, instead of six kernel

types, means that data copy must be done to provide the format of the one conventional kernel. This means a certain performance loss that might have to be repeated several times. The six possible kernels have different pre-fetching patterns.